

On Resilient Graph Spanners[★]

Giorgio Ausiello¹, Paolo G. Franciosa²,
Giuseppe F. Italiano³, and Andrea Ribichini¹

¹ Dipartimento di Ingegneria Informatica, Automatica e Gestionale, Università di Roma “La Sapienza”, via Ariosto 25, 00185 Roma, Italy,
e-mail: {ausiello, ribichini}@dis.uniroma1.it

² Dipartimento di Scienze Statistiche, Università di Roma “La Sapienza”, piazzale Aldo Moro 5, 00185 Roma, Italy, e-mail: paolo.franciosa@uniroma1.it

³ Dipartimento di Ingegneria Civile e Ingegneria Informatica, Università di Roma “Tor Vergata”, via del Politecnico 1, 00133 Roma, Italy,
e-mail: italiano@disp.uniroma2.it

Abstract. We introduce and investigate a new notion of resilience in graph spanners. Let S be a spanner of a graph G . Roughly speaking, we say that a spanner S is resilient if all its point-to-point distances are resilient to edge failures. Namely, whenever any edge in G fails, then as a consequence of this failure all distances do not degrade in S substantially more than in G (i.e., the relative distance increases in S are very close to those in the underlying graph G). In this paper we show that sparse resilient spanners exist, and that they can be computed efficiently.

1 Introduction

Spanners are fundamental graph structures that have been extensively studied in the last three decades. Given a graph G , a *spanner* is a (sparse) subgraph of G that preserves the approximate distance between each pair of vertices. More precisely, for $\alpha \geq 1$ and $\beta \geq 0$, an (α, β) -spanner of a graph $G = (V, E)$ is a subgraph $S = (V, E_S)$, $E_S \subseteq E$, that distorts distances in G up to a multiplicative factor α and an additive term β : i.e., for all vertices x, y , $d_S(x, y) \leq \alpha \cdot d_G(x, y) + \beta$, where d_G denotes the distance in graph G . We refer to (α, β) as the *distorsion* of the spanner. As a special case, $(\alpha, 0)$ -spanners are known as *multiplicative spanners* (also denoted as t -spanners, for $t = \alpha$: $d_S(x, y) \leq t \cdot d_G(x, y)$), and $(1, \beta)$ -spanners are known as *additive spanners* ($d_S(x, y) \leq d_G(x, y) + \beta$). Note that an (α, β) -spanner is trivially a multiplicative $(\alpha + \beta)$ -spanner. It is known how to compute in $O(m + n)$ time a multiplicative $(2k - 1)$ -spanner, with $O(n^{1+\frac{1}{k}})$ edges [2,16] (which is conjectured to be optimal for any k), in $O(m\sqrt{n})$ time additive 2-spanners with $O(n^{\frac{3}{2}})$ edges [1] and in $O(mn^{2/3})$ time additive 6-spanners with

[★] In a previous version of this paper, posted in <http://arxiv.org/abs/1303.1559v1>, the notion of *resilient* graph spanner was named *robust*. The name has been changed to avoid conflict with a different notion of robustness in geometric graph spanners, see [10].

$O(n^{\frac{4}{3}})$ edges [7], where m and n are respectively the number of edges and vertices in the original graph G . Multiplicative t -spanners are only considered for $t \geq 3$, as multiplicative 2-spanners can have as many as $\Theta(n^2)$ edges: this implies that (α, β) -spanners are considered for $\alpha + \beta \geq 3$.

Spanners have been investigated also in the fully dynamic setting, where edges may be added to or deleted from the original graph. In [4], a $(2,1)$ -spanner and a $(3,2)$ -spanner of an unweighted graph are maintained under an intermixed sequence of $\Omega(n)$ edge insertions and deletions in $O(\Delta)$ amortized time per operation, where Δ is the maximum vertex degree of the original graph. The $(2,1)$ -spanner has $O(n^{3/2})$ edges, while the $(3,2)$ -spanner has $O(n^{4/3})$ edges. A faster randomized dynamic algorithm for general multiplicative spanners has been later proposed by Baswana [6]: given an unweighted graph, a $(2k-1, 0)$ -spanner of expected size $O(k \cdot n^{1+1/k})$ can be maintained in $O(\frac{m}{n^{1+1/k}} \cdot \text{polylog } n)$ amortized expected time for each edge insertion/deletion, where m is the current number of edges in the graph. For $k = 2, 3$ (multiplicative 3- and 5-spanners), the amortized expected time of the randomized algorithm becomes constant. The algorithm by Elkin [15] maintains a $(2k-1, 0)$ -spanner with expected $O(kn^{1+1/k})$ edges in expected constant time per edge insertion and expected $O(\frac{m}{n^{1/k}})$ time per edge deletion. More recently, Baswana et al. [8] proposed two faster fully dynamic randomized algorithms for maintaining $(2k-1, 0)$ -spanners of unweighted graphs: the update expected time per insertion/deletion is $O(7^{k/2})$ for the first algorithm and $O(k^2 \log^2 n)$ for the second algorithm, and in both cases the spanner expected size is optimal up to a polylogarithmic factor.

As observed in [12], this traditional fully dynamic model may be too pessimistic in several application scenarios, where the possible changes to the underlying graph are rather limited. Indeed, there are cases where there can be only temporary network failures: namely, graph edges may occasionally fail, but only for a short period of time, and it is possible to recover quickly from such failures. In those scenarios, rather than maintaining a fully dynamic spanner, which has to be updated after each change, one may be more interested in working with a static spanner capable of retaining much of its properties during edge deletions, i.e., capable of being resilient to transient failures.

Being inherently sparse, a spanner is not necessarily resilient to edge deletions and it may indeed lose some of its important properties during a transient failure. Indeed, let S be an (α, β) -spanner of G : if an edge e fails in G , then the distortion of the spanner may substantially degrade, i.e., $S \setminus e$ may no longer be an (α, β) -spanner or even a valid spanner of $G \setminus e$, where $G \setminus e$ denotes the graph obtained after removing edge e from G . In their pioneering work, Chechik et al. [12] addressed this problem by introducing the notion of *fault-tolerant spanners*, i.e., spanners that are resilient to edge (or vertex) failures. Given an integer $f \geq 1$, a spanner is said to be f -edge (resp. vertex) fault-tolerant if it preserves its original distortion under the failure of any set of at most f edges (resp. vertices). More formally, an f -edge (resp. vertex) fault-tolerant (α, β) -spanner of $G = (V, E)$ is a subgraph $S = (V, E_S)$, $E_S \subseteq E$, such that for any subset $F \subseteq E$ (resp. $F \subseteq V$), with $|F| \leq f$, and for any pair of vertices $x, y \in V$ (resp. $x, y \in V \setminus F$) we

have $d_{S \setminus F}(x, y) \leq \alpha \cdot d_{G \setminus F}(x, y) + \beta$, where $G \setminus F$ denotes the subgraph of G obtained after deleting the edges (resp. vertices) in F . Algorithms for computing efficiently fault-tolerant spanners can be found in [5,11,12,14].

The distortion is not the only property of a spanner that may degrade because of edge failures. Indeed, even when the removal of an edge cannot change the overall distortion of a spanner (such as in the case of a fault-tolerant spanner), it may still cause a sharp increase in some of its distances. Note that while the distortion is a *global* property, distance increases are *local* properties, as they are defined for pairs of vertices. To address this problem, one would like to work with spanners that are not only *globally resilient* (such as fault-tolerant spanners) but also *locally resilient*. In other terms, we would like to make the distances between any pair of vertices in a spanner resilient to edge failures, i.e., whenever an edge fails, then the increases in distances in the spanner must be very close to the increases in distances in the underlying graph. More formally, given a graph G and an edge e in G , we define the *fragility of edge e* as the maximum relative increase in distance between any two vertices when e is removed from G :

$$\text{frag}_G(e) = \max_{x,y \in V} \left\{ \frac{d_{G \setminus e}(x, y)}{d_G(x, y)} \right\}$$

Our definition of fragility of an edge is somewhat reminiscent of the notion of *shortcut value*, as contained in [19], where the distance increase is alternatively measured by the difference, instead of the ratio, between distances in $G \setminus e$ and in G . Note that for unweighted graphs, $\text{frag}_G(e) \geq 2$ for any edge e . The fragility of edge e can be seen as a measure of how much e is crucial for the distances in G , as it provides an upper bound to the increase in distance in G between any pair of vertices when edge e fails: the higher the fragility of e , the higher is the relative increase in some distance when e is deleted.

Our contribution. To obtain spanners whose distances are resilient to transient edge failures, the fragility of each edge in the spanner must be as close as possible to its fragility in the original graph. In this perspective, we say that a spanner S of G is σ -*resilient* if $\text{frag}_S(e) \leq \max\{\sigma, \text{frag}_G(e)\}$ for each edge $e \in S$, where σ is a positive integer. Note that in case of unweighted graphs, for $\sigma = 2$ this is equivalent to $\text{frag}_S(e) = \text{frag}_G(e)$. We remark that finding sparse 2-resilient spanners may be an overly ambitious goal, as we prove that there exists a family of dense graphs for which the only 2-resilient spanner coincides with the graph itself. It can be easily seen that in general (α, β) -spanners are not σ -resilient. Furthermore, it can be shown that even edge fault-tolerant multiplicative t -spanners are not σ -resilient, since they can only guarantee that the fragility of a spanner edge is at most t times its fragility in the graph. In fact, we exhibit 1-edge fault tolerant t -spanners, for any $t \geq 3$, with edges whose fragility in the spanner is at least $t/2$ times their fragility in G .

It seems quite natural to ask whether sparse σ -resilient spanners exist, and how efficiently they can be computed. We show that it is possible to compute σ -resilient $(2,1)$ -spanners, $(1,2)$ -spanners and $(3,0)$ -spanners of optimal asymptotic size (i.e., containing $O(n^{3/2})$ edges). The total time required to compute our

spanners is $O(mn)$ in the worst case. To compute our σ -resilient spanners, we start from a non-resilient spanner, and then add to it $O(n^{3/2})$ edges from a carefully chosen set of short cycles in the original graph. The algorithm is simple and thus amenable to practical implementation, while the upper bound on the number of added edges is derived from non-trivial combinatorial arguments.

The same approach can be used for turning a given (α, β) -spanner into a σ -resilient (α, β) -spanner, for any $\sigma \geq \alpha + \beta > 3$, by adding $O(n^{3/2})$ edges. Note that this result is quite general, as (α, β) -spanners contain as special cases all $(k, k-1)$ -spanners, all multiplicative $(2k-1)$ -spanners, for $k \geq 2$, and all additive spanners, including additive 2-spanners and 6-spanners.

All our bounds hold for undirected unweighted graphs and can be extended to the case of graphs with positive edge weights.

Our results for $\sigma = \alpha + \beta = 3$ seem to be the most significant ones, both from the theoretical and the practical point of view. From a theoretical perspective, our σ -resilient (α, β) -spanners, with $\alpha + \beta = 3$, have the same asymptotic size as their non-resilient counterparts. From a practical perspective, there is empirical evidence [3] that small stretch spanners provide the best performance in terms of stretch/size trade-offs, and that spanners of larger stretch are not likely to be of practical value.

Table 1 summarizes previous considerations and compares our results with the fragility and size of previously known spanners.

Spanner S	$d_S(x, y)$	$\text{frag}_S(e)$	Size	Ref.
multiplicative $(2k-1)$ -spanner, $k \geq 2$	$\leq (2k-1) \cdot d_G(x, y)$	unbounded	$O\left(n^{1+\frac{1}{k}}\right)$	[2]
additive 2-spanner	$\leq d_G(x, y) + 2$	unbounded	$O\left(n^{\frac{3}{2}}\right)$	[1]
additive 6-spanner	$\leq d_G(x, y) + 6$	unbounded	$O\left(n^{\frac{4}{3}}\right)$	[7]
1-edge fault-tolerant $(2k-1)$ -spanner, $k \geq 2$	$\leq (2k-1) \cdot d_G(x, y)$	$\leq (2k-1) \cdot \text{frag}_G(e)$	$O\left(n^{1+\frac{1}{k}}\right)$	[12]
σ -resilient (α, β) -spanner, $\sigma \geq \alpha + \beta \geq 3$	$\leq \alpha \cdot d_G(x, y) + \beta$	$\leq \max\{\sigma, \text{frag}_G(e)\}$	$O\left(n^{\frac{3}{2}}\right)$	this paper

Table 1. Fragility and size of spanners.

2 Preliminaries

Let $G = (V, E)$ be an undirected unweighted graph, with m edges and n vertices. The *girth* of G , denoted by $\text{girth}(G)$, is the length of a shortest cycle in G . A *bridge* is an edge $e \in E$ whose deletion increases the number of connected components of G . Note that an edge is a bridge if and only if it is not contained in any cycle of G . Graph G is *2-edge-connected* if it does not have any bridges. The *2-edge-connected components* of G are its maximal 2-edge-connected subgraphs. Let $e \in E$, and denote by \mathcal{C}_e the set of all the cycles containing edge e : if G is 2-edge-connected, then \mathcal{C}_e is non-empty for each $e \in E$. A shortest cycle among all cycles in \mathcal{C}_e is referred to as a *short cycle for edge e* . If G is 2-edge-connected

short cycles always exist for any edge. Short cycles are not necessarily unique: for each $e \in E$, we denote by Γ_e the set of short cycles for e . Similarly, we denote by $\mathcal{P}_e(x, y)$ the set of all paths between x and y containing edge e , and by $\mathcal{P}_{\bar{e}}(x, y)$ the set of all paths between x and y avoiding edge e . We further denote by $\Pi_e(x, y)$ (respectively $\Pi_{\bar{e}}(x, y)$) the set of shortest paths in $\mathcal{P}_e(x, y)$ (respectively $\mathcal{P}_{\bar{e}}(x, y)$).

Recall that we defined the *fragility* of an edge $e = (u, v)$ in graph G as $\text{frag}_G(e) = \max_{x, y \in V} \left\{ \frac{d_{G \setminus e}(x, y)}{d_G(x, y)} \right\}$. The following lemma shows that in this definition the maximum is obtained for $\{x, y\} = \{u, v\}$, i.e., exactly at the two endpoints of edge (u, v) .

Lemma 1. *Let $G = (V, E)$ be a connected graph with positive edge weights, and let $e = (u, v)$ be any edge in G . Then $\text{frag}_G(e) = \frac{d_{G \setminus e}(u, v)}{d_G(u, v)}$.*

Proof. Let x and y be any two vertices in G . To prove the lemma it suffices to show that $\frac{d_{G \setminus e}(x, y)}{d_G(x, y)} \leq \frac{d_{G \setminus e}(u, v)}{d_G(u, v)}$. We distinguish two cases, depending on whether there is a shortest path in G between x and y that avoids edge e or not. If there is such a shortest path, then $d_{G \setminus e}(x, y) = d_G(x, y)$. Since $d_{G \setminus e}(u, v) \geq d_G(u, v)$, the lemma trivially holds.

Assume now that all shortest paths between x and y in G go through the edge $e = (u, v)$. In this case, $d_G(x, y) \geq d_G(u, v)$. If edge e is a bridge, then $\text{frag}_G(e) = \frac{d_{G \setminus e}(u, v)}{d_G(u, v)} = +\infty$, and again the lemma holds trivially. If e is not a bridge, then the graph $G \setminus e$ is connected. Since there is at least a (not necessarily shortest) path in $G \setminus e$ between x and y containing the shortest path in $G \setminus e$ from u to v , we have that $d_{G \setminus e}(x, y) \leq d_G(x, y) - d_G(u, v) + d_{G \setminus e}(u, v)$, or equivalently

$$\frac{d_{G \setminus e}(x, y)}{d_G(x, y)} \leq \frac{d_G(x, y) - d_G(u, v) + d_{G \setminus e}(u, v)}{d_G(x, y)} \quad (1)$$

Since $d_G(x, y) - d_G(u, v) + d_{G \setminus e}(u, v) \geq d_G(x, y)$, we can upper bound the right-hand side of (1) by subtracting $d_G(x, y) - d_G(u, v) \geq 0$ to both its numerator and denominator, yielding the lemma. \square

Note that for unweighted graphs, Lemma 1 can be stated as $\text{frag}_G(e) = d_{G \setminus e}(u, v)$.

The fragility of all edges in a graph $G = (V, E)$ with positive edge weights can be trivially computed in a total of $O(m^2n + mn^2 \log n)$ worst-case time by simply computing all-pairs shortest paths in all graphs $G \setminus e$, for each edge $e \in E$. A faster bound of $O(mn + n^2 \log n)$ can be achieved by using either a careful modification of algorithm **fast-exclude** in [13] or by applying n times a modified version of Dijkstra's algorithm, as described in [17]. For unweighted graphs, the above bound reduces to $O(mn)$.

3 Computing σ -resilient subgraphs

We first show that finding sparse 2-resilient spanners may be an ambitious goal, as there are dense graphs for which the only 2-resilient spanner is the graph itself.

Theorem 1. *There is an infinite family \mathcal{F} of graphs such that for each graph $G \in \mathcal{F}$ the following properties hold:*

- (1) *G has $\Theta(n^\delta)$ edges, with $\delta > 1.72598$, where n is the number of vertices of G .*
- (2) *No proper subgraph of G is a 2-resilient spanner of G .*
- (3) *There exists a 2-spanner S of G such that $\Theta(n^\delta)$ edges of $G \setminus S$, with $\delta > 1.72598$, need to be added back to S in order to make it 2-resilient.*

Proof. The family \mathcal{F} is defined as the set of graphs $\{I_3, I_6, I_9, \dots, I_{3k}, \dots\}$, with each I_{3k} being the complement of the intersection graph of all the k -sets contained in a $3k$ -set, $k \geq 1$. Given a set U , with $|U| = 3k$, graph I_{3k} contains a vertex v_A for each subset $A \subset U$ with $|A| = k$, and vertex v_A is adjacent to vertex v_B if and only if $A \cap B = \emptyset$.

Graph I_{3k} has $\binom{3k}{k}$ vertices, each having degree $\binom{2k}{k}$, since this is the number of k -sets that can be chosen from the remaining $2k$ elements. Let $n = \binom{3k}{k}$ be the number of vertices in I_{3k} : then I_{3k} has $m = \frac{n}{2} \binom{2k}{k}$ edges. By Stirling approximation, we have that $m = \Theta\left(n^{1 + \frac{2}{3 \log_2 3 - 2}}\right)$, where $\frac{2}{3 \log_2 3 - 2} > 0.72598$. This proves Property (1).

We now turn to Property (2). We first claim that there is only one path of length 2 between any pair of adjacent vertices in I_{3k} . Indeed, for any two adjacent vertices v_A and v_B , there is exactly one vertex, namely $v_{U \setminus (A \cup B)}$, which is adjacent to both v_A and v_B . Thus each edge belongs to exactly one triangle, which implies that the fragility of any edge in I_{3k} is 2, and that there is only one path of length 2 between any pair of adjacent vertices. Let $S \subset I_{3k}$ be a 2-spanner of I_{3k} . We show that S is not 2-resilient. Let v_A and v_B be two adjacent vertices in I_{3k} such that $(v_A, v_B) \notin S$, and let $C = U \setminus (A \cup B)$. We know that v_A, v_C, v_B is the only path of length 2 from v_A to v_B in I_{3k} . Since S is a 2-spanner of I_{3k} , both (v_A, v_C) and (v_C, v_B) must be in S . For the same reason above, the only path of length 2 in I_{3k} from v_A to v_C is v_A, v_B, v_C , so $d_{S \setminus \{(v_A, v_C)\}}(v_A, v_C) > 2$, because $(v_A, v_B) \notin S$. Thus $\text{frag}_S((v_A, v_C)) > 2$, while $\text{frag}_{I_{3k}}((v_A, v_C)) = 2$, which implies that S is not 2-resilient.

To prove Property (3), let S be a subgraph of I_{3k} obtained by deleting exactly one edge from each triangle in I_{3k} . Since each edge in I_{3k} is contained in exactly one triangle, there is always such an S , and it is a 2-spanner of I_{3k} . Furthermore, by Property (1), $\frac{m}{3} = \Theta(n^\delta)$ edges, with $\delta > 1.72598$, have to be deleted from I_{3k} in order to produce S . By Property (2), $\Theta(n^\delta)$ edges of $I_{3k} \setminus S$, with $\delta > 1.72598$, need to be added back to S in order to make it a 2-resilient 2-spanner of I_{3k} . \square

Edge fault-tolerant spanners provide a simple way to bound distance increases under edge faults. Unfortunately, they are not σ -resilient, as the next lemma shows.

Lemma 2. *Let $G = (V, E)$ be a graph.*

- (a) *Let S_f be any 1-edge fault tolerant t -spanner of G . Then $\text{frag}_{S_f}(e) \leq t \cdot \text{frag}_G(e)$ for each $e \in S_f$.*

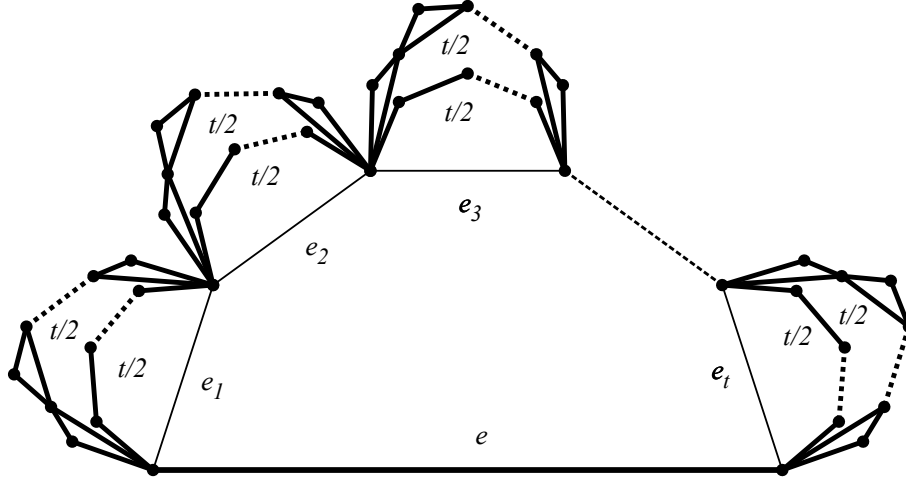


Fig. 1. A 1-edge fault tolerant t -spanner that is not σ -resilient for any $\sigma < t/2$. Edges e_i , $1 \leq i \leq t$, are not included in the t -spanner.

(b) *There exist 1-edge fault-tolerant t -spanners that are not σ -resilient, for any $\sigma < t/2$.*

Proof. We first prove (a). By definition of 1-edge fault tolerant t -spanner, we have $d_{S_f \setminus e}(u, v) \leq t \cdot d_{G \setminus e}(u, v)$ for each edge $e = (u, v)$, and since $e \in S_f$ we have $d_{S_f}(u, v) = d_G(u, v)$. The fragility of e in S_f is then:

$$\text{frag}_{S_f}(e) = \frac{d_{S_f \setminus e}(u, v)}{d_{S_f}(u, v)} \leq \frac{t \cdot d_{G \setminus e}(u, v)}{d_{S_f}(u, v)} = \frac{t \cdot d_{G \setminus e}(u, v)}{d_G(u, v)} = t \cdot \text{frag}_G(e).$$

To prove (b), consider the graph illustrated in Figure 1. The subgraph defined by bold edges (i.e., the whole graph except edges e_i , with $1 \leq i \leq t$) is a 1-edge fault tolerant t -spanner. The fragility of edge e in the original graph is t , while its fragility in the spanner is $t^2/2$, i.e., it is greater than the fragility in the original graph by a factor of $t/2$. \square

To compute a σ -resilient spanner R of graph G , without any guarantees on the number of edges in R , we may start from any (α, β) -spanner of G , with $\alpha + \beta \leq \sigma$, and add a suitable set of *backup paths* for edges with high fragility:

1. Let S be any (α, β) -spanner of G , with $\alpha + \beta \leq \sigma$: initialize R to S .
2. For each edge $e = (u, v) \in S$ such that $\text{frag}_S(e) > \sigma$, select a shortest path between u and v in $G \setminus e$ and add it to R .

The correctness of our approach hinges on the following theorem.

Theorem 2. *Let S be an (α, β) -spanner of a graph G , and let R be computed by adding to S a backup path for each edge e with $\text{frag}_S(e) > \sigma$. Then R is a σ -resilient (α, β) -spanner of G .*

Proof. R is trivially an (α, β) -spanner, since it contains an (α, β) -spanner S . It remains to show that R is σ -resilient. Let $e = (u, v)$ be any edge in R . We distinguish two cases, depending on whether e was in the initial (α, β) -spanner S or not:

- $e \in S$: if $\text{frag}_S(e) \leq \sigma$ then also $\text{frag}_R(e) \leq \sigma$. If $\text{frag}_S(e) > \sigma$, a shortest path in $G \setminus e$ joining u and v has been added to S , yielding $\text{frag}_R(e) = \text{frag}_G(e)$;
- $e \in R \setminus S$: since S is an (α, β) -spanner of G , it must also contain a path between u and v of length at most $\alpha + \beta$. This implies that $\text{frag}_R(e) \leq \text{frag}_S(e) \leq \alpha + \beta \leq \sigma$. \square

Note that any σ -resilient spanner R , computed adding backup paths for high fragility edges, inherits the properties of the underlying (α, β) -spanner S , i.e., if S is fault-tolerant then R is fault-tolerant too. Let $T(m, n)$ and $S(n)$ be respectively the time required to compute an (α, β) -spanner S and the number of edges in S . A trivial implementation of the above algorithm requires a total of $O(T(m, n) + S(n) \cdot (m + n))$ time for unweighted graphs and produces σ -resilient spanners with $O(n \cdot S(n))$ edges. In the next section we will show how to improve the time complexity and how to limit the number of added edges.

3.1 Main results: improving size and running time

Theorem 2 does not depend on how backup paths are selected. In order to bound the number of added edges, we first show that in an unweighted graph the number of edges with high fragility is small (Theorem 3), and then we show how to carefully select shortest paths to be added as backup paths, so that the total number of additional edges required is small (Theorem 4). By combining the two bounds above, we obtain σ -resilient (α, β) -spanners with $O(n^{\frac{3}{2}})$ edges in the worst case (Theorem 5). We start by bounding the number of high fragility edges in any graph. For lack of space, the following theorem is proved only for unweighted graphs. However, it holds for graphs with positive edge weights as well.

Theorem 3. *Let $G = (V, E)$ be a graph, and let σ be any positive integer. Then, the number of edges of G having fragility greater than σ is $O(n^{1+1/\lfloor(\sigma+1)/2\rfloor})$.*

Proof. Let L be the subgraph of G containing only edges whose fragility is greater than σ . If L contains no cycle, then L has at most $(n-1)$ edges and the theorem trivially follows. Otherwise, let C be a cycle in L , let ℓ be the number of edges in C , and let e be any edge in C . Note that $\text{frag}_L(e) \leq \ell-1$. Since L is a subgraph of G , we have $\text{frag}_G(e) \leq \text{frag}_L(e)$. Thus, $\ell \geq \text{frag}_G(e) + 1$. This holds for any cycle in L , and hence $\text{girth}(L) = \min_{e \in L} \{\text{frag}_L(e)\} + 1 \geq \min_{e \in L} \{\text{frag}_G(e)\} + 1 > \sigma + 1$. As proved by Bondy and Simonovits [9], a graph with girth greater than $\sigma + 1$ contains $O(n^{1+1/\lfloor(\sigma+1)/2\rfloor})$ edges. \square

We now tackle the problem of selecting the shortest paths to be added as backup paths, so that the total number of additional edges is small. Without loss of generality, we assume that G is 2-edge-connected: if it is not, all bridges in G

will necessarily be included in any spanner and our algorithm can be separately applied to each 2-edge-connected component of G . Let $e = (u, v)$ be an edge of high fragility in the initial (α, β) -spanner. Note that, in order to identify a backup path for edge e , we can either refer to a shortest path between u and v in $G \setminus e$ or, equivalently, to a short cycle for e in G (i.e., the short cycle defined by the shortest path in $G \setminus e$ and the edge e itself). In the following, we will use short cycles in G rather than shortest paths in $G \setminus e$. Recall from Section 2 that we denote by Γ_e the set of short cycles for e , and by $\Pi_e(x, y)$ (respectively $\Pi_{\bar{e}}(x, y)$) the set of shortest paths in $\mathcal{P}_e(x, y)$ (respectively $\mathcal{P}_{\bar{e}}(x, y)$). The following property is immediate:

Property 1. Let e be any edge of G , let $C \in \Gamma_e$ be a short cycle for e , and let x, y be any two vertices in C . Then x and y split C into two paths C_e and $C_{\bar{e}}$, with $C_e \in \Pi_e(x, y)$ and $C_{\bar{e}} \in \Pi_{\bar{e}}(x, y)$.

Property 1 allows us to prove the following lemma:

Lemma 3. *Let e and f be any two edges in G . If two short cycles $A \in \Gamma_e$ and $B \in \Gamma_f$ share two common vertices, say x and y , then either $B_f \cup A_e$ or $B_f \cup A_{\bar{e}}$ is a short cycle for edge f , where $\{A_e, A_{\bar{e}}\}$ and $\{B_f, B_{\bar{f}}\}$ are the decompositions of A and B with respect to x and y defined in Property 1.*

Proof. Since A is a short cycle in Γ_e , edge f cannot belong to both A_e and $A_{\bar{e}}$. We distinguish two cases:

- $f \notin A_e$: in this case, $A_e \in \Pi_{\bar{f}}(x, y)$ and replacing $B_{\bar{f}}$ by A_e yields a short cycle in Γ_f .
- $f \in A_e$, which implies that $f \notin A_{\bar{e}}$. In this case, $A_{\bar{e}} \in \Pi_{\bar{f}}(x, y)$ and replacing $B_{\bar{f}}$ by $A_{\bar{e}}$ yields a short cycle in Γ_f . \square

Lemma 3 can be intuitively read as follows. Given two edges e and f , let C_e be a short cycle for e and let C_f be a short cycle for f . If C_e and C_f cross in two vertices, then we can compute an alternative short cycle for f , say C'_f , which has a larger intersection with C_e (i.e., such that $C_e \cup C'_f$ has fewer edges than $C_e \cup C_f$). This property allows us to select backup paths in such a way that the total number of additional edges required is relatively small. To do that efficiently, we apply a modified version of algorithm **fast-exclude** in [13]. For lack of space, we only sketch here the main modifications needed and refer to the full paper for the low-level details of the method. Algorithm **fast-exclude** is based on Dijkstra's algorithm and compute shortest paths avoiding a set of *independent paths*, from a source vertex u to any other vertex. In order to find short cycles for the set F_u of high fragility edges incident to vertex u , we would like to apply algorithm **fast-exclude** so that it avoids all edges in F_u . Unfortunately, F_u is not a proper set of independent paths (as defined in [13]), and so algorithm **fast-exclude** cannot be applied directly. We circumvent this problem by splitting each edge $(u, v) \in F_u$ into two edges (u, v') , (v', v) with the help of an extra vertex v' , and by letting algorithm **fast-exclude** avoid all edges of the form (v', v) , since they form a set of independent paths. A second modification of algorithm **fast-exclude** consists of giving higher priority, during the

Dijkstra-like visits, to the edges that have been already used, so that whenever a short cycle for an edge e has to be output, the algorithm selects a short cycle containing fewer new edges (i.e., edges not already contained in previously output short cycles), as suggested by Lemma 3. This allows us to prove the following theorem.

Theorem 4. *Given $q > 0$ edges e_1, e_2, \dots, e_q in a 2-edge-connected graph G , there always exist short cycles C_1, C_2, \dots, C_q in G , with $C_i \in \Gamma_{e_i}$ for $1 \leq i \leq q$, such that the graph $\cup_{i=1}^q C_i$ has $O(\min\{q\sqrt{n} + n, n\sqrt{q} + q\})$ edges.*

Proof. Let C_1, C_2, \dots, C_q be the cycles in the order in which they are found by the modified version of algorithm **fast-exclude**, and let V_i and E_i be respectively the vertex set and the edge set of C_i , for $1 \leq i \leq q$. We partition each E_i into the following three disjoint sets:

- E_i^{old} : edges in $E_i \cap \left(\bigcup_{j=1}^{i-1} E_j\right)$, i.e., edges already in some E_j , $j < i$.
- E_i^{new} : edges with at least one endpoint not contained in $\bigcup_{j=1}^{i-1} V_j$.
- E_i^{cross} : edges not contained in E_i^{old} and with both endpoints in $\bigcup_{j=1}^{i-1} V_j$.

To prove the theorem, we have to bound the number of edges in $\bigcup_{i=1}^q E_i$. We only need to count the total number of edges in $\bigcup_{i=1}^q E_i^{\text{new}}$ and $\bigcup_{i=1}^q E_i^{\text{cross}}$, since each edge in E_i^{old} , for any $1 \leq i \leq q$, has been already accounted for in some E_j^{new} or E_j^{cross} , with $j < i$. Since each edge in E_i^{new} can be amortized against a new vertex, and at most two new edges are incident to each new vertex, $|\bigcup_{i=1}^q E_i^{\text{new}}| \leq 2 \cdot n$.

To bound the size of sets E_i^{cross} , we proceed as follows. For each cycle C_i we choose an arbitrary orientation \vec{C}_i , in one of the two possible directions and direct its edges accordingly. For directed edge $e = (x, y)$ we denote vertex x as $\text{tail}(e)$. We build a bipartite graph \mathcal{B} in which one vertex class represents the n vertices v_1, v_2, \dots, v_n in G , and the other vertex class represents the q directed short cycles $\vec{C}_1, \vec{C}_2, \dots, \vec{C}_q$. There is an edge in \mathcal{B} joining cycle C_i and vertex v if and only if v is the tail of an edge in E_i^{cross} . It is possible to see that the degree of C_i in \mathcal{B} is the size of E_i^{cross} , since each edge in \mathcal{B} corresponds to an edge in $\bigcup_{i=1}^q E_i^{\text{cross}}$ and vice versa.

We claim that two vertices x and y cannot be tails of two pairs of directed edges in E_i^{cross} and E_j^{cross} (see Figure 2). We prove this claim by contradiction. Assume without loss of generality $i > j$ (i.e., short cycle C_j has been output before short cycle C_i). Since whenever a short cycle has to be output, our algorithm selects a short cycle containing fewer new edges (i.e., edges not contained already in previously output short cycles), either the path τ_1 or τ_2 of C_i should have been replaced by portion π_1 or portion π_2 of C_j (as in Lemma 3). Let f and g be the two directed edges in E_i^{cross} with $\text{tail}(f) = x$ and $\text{tail}(g) = y$: by the above argument, one among f and g should be in E_i^{old} instead of E_i^{cross} , yielding a contradiction.

The previous claim implies that the bipartite graph \mathcal{B} does not contain $K_{2,2}$ as a (not necessarily induced) subgraph. Determining the maximum number of edges in \mathcal{B} is a special case of Zarankiewicz's problem [22]. This problem has

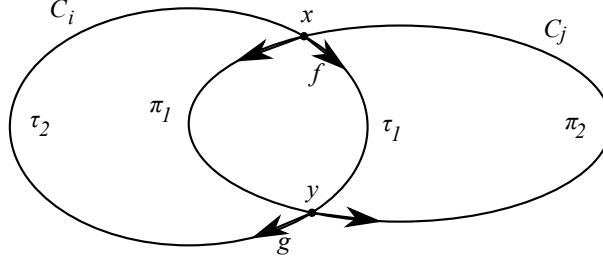


Fig. 2. On the proof of Theorem 4. If cycle C_i is detected after cycle C_j , then either edge f or g is not in E_i^{cross} . In fact, one among π_1 or π_2 should have been included in C_i in place of τ_1 .

been solved by Kővári, Sós, Turán [18] (see also [20], p. 65), who proved that any bipartite graph G with vertex classes of size m and n containing no subgraph $K_{r,s}$, with the r vertices in the class of size m and the s vertices in the class of size n , has $O(\min\{mn^{1-1/r} + n, m^{1-1/s}n + m\})$ edges, where the constant of proportionality depends on r and s . Since in our case the bipartite graph \mathcal{B} has vertex classes of size n and q , and $r = s = 2$, it follows that \mathcal{B} contains $O(\min\{q\sqrt{n} + n, n\sqrt{q} + q\})$ edges.

In summary, the total number of edges in the graph $\bigcup_{i=1}^q C_i$ is bounded by

$$\left| \bigcup_{i=1}^q (E_i^{\text{old}} \cup E_i^{\text{new}} \cup E_i^{\text{cross}}) \right| \leq 2n + O(\min\{q\sqrt{n} + n, n\sqrt{q} + q\})$$

and thus the theorem holds. \square

We are now ready to bound the size and the time required to compute a σ -resilient (α, β) -spanner.

Theorem 5. *Let G be a graph with m edges and n vertices. Let S be any (α, β) -spanner of G , and denote by $T(m, n)$ and $S(n)$ respectively the time required to compute S and the number of edges in S . Then a σ -resilient (α, β) -spanner R of G , with $\sigma \geq \alpha + \beta$, can be computed in $O(T(n) + mn)$ time. Furthermore, $R \supseteq S$ and R has $O(S(n) + n^{3/2})$ edges.*

Proof. As explained above, a σ -resilient (α, β) -spanner R can be computed by adding a set \mathcal{C} of short cycles to S , one for each edge $e \in S$ with $\text{frag}_G(e) > \sigma$. Let C_e be the cycle in Γ_e computed by our algorithm.

We partition edges $e \in S$ with $\text{frag}_G(e) > \sigma$ into three subsets E_ℓ , E_m and E_h , according to their fragility in G . For each subset we separately bound the number of edges in the union of cycles in \mathcal{C} .

low fragility edges: $E_\ell = \{e \in S \mid \sigma \leq \text{frag}_G(e) \leq 5\}$. Obviously, we have $|E_\ell| \leq S(n)$, and since each cycle C_e , $e \in E_\ell$, contains at most 5 edges

$$\left| \bigcup_{e \in E_\ell} C_e \right| = O(S(n))$$

medium fragility edges: $E_m = \{e \in S \mid \max\{\sigma, 6\} \leq \text{frag}_G(e) < \log n\}$. By Theorem 3, since the fragility of each edge in E_m is greater than 5, $|E_m| = O(n^{4/3})$. Since each cycle C_e , $e \in E_m$, contains at most $\log n$ edges

$$\left| \bigcup_{e \in E_m} C_e \right| = O\left(n^{\frac{4}{3}} \cdot \log n\right)$$

high fragility edges: $E_h = \{e \in S \mid \text{frag}_G(e) \geq \log n\}$. By Theorem 3, $|E_h| = O\left(n^{1+\frac{2}{\log n}}\right) = O(n)$, and by Theorem 4 we have that

$$\left| \bigcup_{e \in E_h} C_e \right| = O\left(n \cdot \sqrt{|E_h|}\right) = O\left(n^{\frac{3}{2}}\right)$$

Hence the total number of edges in R is

$$\left| \bigcup_{e \in E_t \cup E_m \cup E_h} C_e \right| = O\left(S(n) + n^{\frac{3}{2}}\right)$$

To bound the running time, we observe that we find the fragility of each edge in S and then we compute a set of short cycles as suggested by Lemma 3. The fragility of each edge and the set of short cycles can be computed by a proper modification of algorithm **fast-exclude** in [13] in a total of $O(mn)$ worst-case time. \square

Theorem 5 allows us to compute σ -resilient versions of several categories of spanners, including multiplicative $(2k-1)$ -spanners and $(k, k-1)$ -spanners for $k \geq 2$, $(1, 2)$ -spanners and $(1, 6)$ -spanners. Since the time required to compute all those underlying spanners is $o(mn)$, in all those cases the time required to build a σ -resilient spanner is $O(mn)$. Theorem 5 can also be applied to build σ -resilient f -spanners, where f is a general *distortion* function as defined in [21], provided that $\sigma \geq f(1)$. Furthermore, if we wish to compute a σ -resilient (α, β) -spanner with $\sigma < \alpha + \beta$, the same algorithm can still be applied starting from a σ -spanner instead of an (α, β) -spanner, yielding the same bounds given in Theorem 5.

Our results can be extended to weighted graphs, since Theorems 3 and 4 also hold for graphs with positive edge weights. Let w_{max} and w_{min} be respectively the weights of the heaviest and lightest edge in the graph, and let $W = \frac{w_{max}}{w_{min}}$. For either $\sigma > \log n$ or $\sigma \geq 5$ and $W = O\left(\left(n^{\frac{1}{2} - \frac{1}{\lfloor(\sigma+1)/2\rfloor}}\right) / \log n\right)$, we can compute a σ -resilient t -spanner with $O(n^{\frac{3}{2}})$ edges in $O(mn)$ time. In the remaining cases (either $\sigma \geq 5$ and larger W or $\sigma = 3, 4$), the total number of edges becomes $O(W \cdot n^{\frac{3}{2}})$. For lack of space, the details are deferred to the full paper.

4 Conclusions and further work

In this paper, we have investigated a new notion of resilience in graph spanners by introducing the concept of σ -resilient spanners. In particular, we have shown that

it is possible to compute small stretch σ -resilient spanners of optimal size. The techniques introduced for small stretch σ -resilient spanners can be used to turn any generic (α, β) -spanner into a σ -resilient (α, β) -spanner, for $\sigma \geq \alpha + \beta > 3$, by adding a suitably chosen set of at most $O(n^{3/2})$ edges. The same approach is also valid for graphs with positive edge weights.

We expect that in practice our σ -resilient spanners, for $\sigma \geq \alpha + \beta > 3$, will be substantially sparser than what it is implied by the bounds given in Theorem 5, and thus of higher value in applicative scenarios. Towards this aim, we plan to perform a thorough experimental study. Another intriguing question is whether our theoretical analysis on the number of edges that need to be added to an (α, β) -spanner in order to make it σ -resilient provides tight bounds, or whether it can be further improved.

References

1. D. Aingworth, C. Chekuri, P. Indyk, and R. Motwani. Fast estimation of diameter and shortest paths (without matrix multiplication). *SIAM J. Comput.*, 28(4):1167–1181, 1999.
2. I. Althofer, G. Das, D. P. Dobkin, D. Joseph, and J. Soares. On sparse spanners of weighted graphs. *Discrete & Computational Geometry*, 9:81–100, 1993.
3. G. Ausiello, C. Demetrescu, P. G. Franciosa, G. F. Italiano, and A. Ribichini. Graph spanners in the streaming model: An experimental study. *Algorithmica*, 55(2):346–374, 2009.
4. G. Ausiello, P. G. Franciosa, and G. F. Italiano. Small stretch spanners on dynamic graphs. *Journal of Graph Algorithms and Applications*, 10(2):365–385, 2006.
5. G. Ausiello, P. G. Franciosa, G. F. Italiano, and A. Ribichini. Computing graph spanner in small memory: fault-tolerance and streaming. *Discrete Mathematics, Algorithms and Applications*, 2(4):591–605, 2010.
6. S. Baswana. Dynamic algorithms for graph spanners. In *Proc. of 13th Annual European Symposium on Algorithms (ESA’06)*, pages 76–87, 2006.
7. S. Baswana, T. Kavitha, K. Mehlhorn, and S. Pettie. New constructions of (α, β) -spanners and purely additive spanners. In *Proc. of the 16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA’05)*, pages 672–681, 2005.
8. S. Baswana, S. Khurana, and S. Sarkar. Fully dynamic randomized algorithms for graph spanners. *ACM Trans. Algorithms*, 8(4):35:1–35:51, October 2012.
9. J.A. Bondy and M. Simonovits. Cycles of even length in graphs. *Journal of Combinatorial Theory, Series B*, 16(2):97–105, 1974.
10. P. Bose, V. Dujmovic, P. Morin, and M. Smid. Robust geometric spanners. In *Symposium on Computational Geometry (SoCG 2013)*, 2013. To appear.
11. G. Braunschvig, S. Chechik, and D. Peleg. Fault tolerant additive spanners. In *Graph-Theoretic Concepts in Computer Science - 38th International Workshop, (WG’12)*, volume 7551 of *LNCS*, pages 206–214. Springer, 2012.
12. S. Chechik, M. Langberg, D. Peleg, and L. Roditty. Fault-tolerant spanners for general graphs. In *Proc. of 41st Annual ACM Symposium on Theory of Computing (STOC’09)*, pages 435–444, 2009.
13. C. Demetrescu, M. Thorup, R.A. Chowdhury, and V. Ramachandran. Oracles for distances avoiding a failed node or link. *SIAM J. Comput.*, 37(5):1299–1318, 2008.

14. M. Dinitz and R. Krauthgamer. Fault-tolerant spanners: better and simpler. In *Proc. of the 30th Annual ACM Symposium on Principles of Distributed Computing (PODC'11)*, pages 169–178, 2011.
15. M. Elkin. Streaming and fully dynamic centralized algorithms for constructing and maintaining sparse spanners. In *Proc. of the 34th International Colloquium on Automata, Languages and Programming (ICALP'07)*, volume 4596 of *LNCS*, pages 716–727. Springer, 2007.
16. S. Halperin and U. Zwick. Linear time deterministic algorithm for computing spanners for unweighted graphs. Unpublished manuscript, 1996.
17. R. Jacob, D. Koschützki, K.A. Lehmann, L. Peeters, and D. Tenfelde-Podehl. Algorithms for centrality indices. In U. Brandes and T. Erlebach, editors, *Network Analysis*, volume 3418 of *LNCS*, pages 62–82. Springer, 2005.
18. T. Kővári, V. T. Sós, and P. Turán. On a problem of K. Zarankiewicz. *Colloquium Mathematicae*, 3(1):50–57, 1954.
19. D. Koschützki, K.A. Lehmann, L. Peeters, S. Richter, D. Tenfelde-Podehl, and O. Zlotowski. Centrality indices. In U. Brandes and T. Erlebach, editors, *Network Analysis*, volume 3418 of *LNCS*, pages 16–61. Springer, 2005.
20. J. Matoušek. *Lectures on Discrete Geometry*. Springer, 2002.
21. S. Pettie. Low distortion spanners. In *Proc. of the 34th International Colloquium on Automata, Languages and Programming (ICALP'07)*, volume 4596 of *LNCS*, pages 78–89. Springer, 2007.
22. K. Zarankiewicz. Problem p 101. *Colloquium Mathematicae*, 2:301, 1951.